Remarks

Entry of the amendments, reconsideration of the application, as amended, and allowance of all pending claims are respectfully requested. Claims 1, 11, 21 and 22 have been amended to further define that the calling program and callee program coexist within a single executable module but have different machine context organizations. Support for this amendment can be found throughout the specification (e.g., page 22, lines 15-17). Thus, no new matter is being added. Claims 1-40 remain pending.

Pursuant to 37 C.F.R. 1.121(c)(1)(ii), a marked-up version of the amended claims is provided on one or more pages separate from the amendment. These pages are appended at the end of the Response.

In the Office Action dated October 24, 2002, claims 1-40 are rejected under 35 U.S.C. 102(b) as being anticipated by Breslau et al. (U.S. Patent No. 5,774,728). Applicants respectfully, but most strenuously, traverse this rejection to any extent deemed applicable to the amended claims presented herewith.

In accordance with an aspect of the present invention, a capability is provided in which programs having different machine context organizations can communicate with one another. For example, communication between a calling program having one machine context organization and a callee program having another machine context organization is mediated by a linkage service, which enables those programs to coexist within a single executable module.

As one particular example, applicants claim a method of communicating between programs having different machine context organizations (e.g., claim 1). The method includes, for instance, determining, at compile time, which savearea layout of a plurality of savearea layouts is to be used to save information relating to a calling program; and selecting, at compile time, a linkage service from a plurality of linkage services to be used in communicating between the calling program and a callee program, wherein the calling program and the callee program coexist within a single executable module but have different machine context organizations.

Thus, in applicants' claimed invention, although a calling program and a callee program have different machine context organizations, they still coexist within a single executable module. This allows, for instance, both programs to be executed in the same execution environment, such as a small architecture environment or a large architecture environment. This is very different from the teachings of Breslau.

Breslau explicitly teaches the generation of multiple executable modules, which is in sharp contrast to applicants' claimed invention, in which the calling program and callee program coexist within a single executable module. For instance, in Breslau at Col. 4, lines 1-21, it describes that at least two sections of source code are intended for different execution environments, and thus, are specifically compiled for those environments. Those specific compilations are then executed in the chosen environments (see, also, Col. 7, lines 5-9). Therefore, in Breslau, there are multiple executable modules and not a single executable module, as claimed by applicants.

In support of the rejection, reference is made in the Office Action to Col. 1, lines 58-67 and Col. 2. However, applicants respectfully submit that these sections do not describe programs having different machine context organizations coexisting in a single executable module. Instead, those sections describe a single source code, rather than a single executable module. That is, Breslau starts with a single source code, but then that source code is compiled into multiple executable modules, which are executed. There is no description, teaching or suggestion in Breslau of enabling programs with different machine context organizations to coexist within a single executable module. As a matter of fact, Breslau teaches away from such a concept.

Based on the foregoing, applicants respectfully submit that Breslau does not anticipate, teach or suggest applicants' claimed invention. Thus, applicants respectfully request an indication of allowability of the independent claims.

The dependent claims are patentable for the same reasons as the independent claims, as well as for their own additional features. For example, in dependent claim 7, applicants recite

that at least two savearea layouts of the plurality of savearea layouts coexist within a single executable module. Applicants respectfully submit that this is not described, taught or suggested in Breslau. Initially, Breslau does not make any mention of savearea layouts. Breslau is using a higher level programming language, and thus, is not directly concerned with savearea layouts. Further, there is no description, teaching or suggestion of a plurality of savearea layouts coexisting within a single executable module. Again, Breslau does not teach a single executable module, but instead a single source code. The single source code is used to produce multiple executable modules. Thus, in Breslau, if savearea layouts were discussed (which applicants are not conceding), each executable module would have its own savearea layout. Again, there is no teaching or suggestion of a plurality of savearea layouts coexisting within a single executable module. Therefore, applicants respectfully submit that claim 7, as well as claims 17 and 28, are patentable over Breslau.

Based on the foregoing, applicants respectfully request an indication of allowability for all pending claims.

Should the Examiner wish to discuss this case with applicants' attorney, please contact applicants' attorney at the below listed number.

Respectfully submitted,

Blanche E. Schiller

Blanche E. Schiller
Attorney for Applicants
Registration No. 35,670

Dated: January 24, 2003

HESLIN ROTHENBERG FARLEY & MESITI P.C.
5 Columbia Circle
Albany, New York 12203
Telephone: (518) 452-5600
Facsimile: (518) 452-5579

Version with markings to show changes made

In the Claims:

Claims 1, 11, 21 and 22 have been amended, as follows:

1. (AMENDED) A method of communicating between programs having different machine context organizations, said method comprising:

determining, at compile time, which savearea layout of a plurality of savearea layouts is to be used to save information relating to a calling program; and

selecting, at compile time, a linkage service from a plurality of linkage services to be used in communicating between said calling program and a callee program, wherein said calling program and said callee program coexist within a single executable module but have different machine context organizations.

11. (AMENDED) A system of communicating between programs having different machine context organizations, said system comprising:

means for determining, at compile time, which savearea layout of a plurality of savearea layouts is to be used to save information relating to a calling program; and

means for selecting, at compile time, a linkage service from a plurality of linkage services to be used in communicating between said calling program and a callee program, wherein said calling program and said callee program coexist within a single executable module but have different machine context organizations.

21. (AMENDED) A system of communicating between programs having different machine context organizations, said system comprising:

a computing environment adapted to determine, at compile time, which savearea layout of a plurality of savearea layouts is to be used to save information relating to a calling program; and

said computing environment being further adapted to select, at compile time, a linkage service from a plurality of linkage services to be used in communicating between said calling program and a callee program, wherein said calling program and said callee program coexist within a single executable module but have different machine context organizations.

22. (AMENDED) At least one program storage device readable by a machine, tangibly embodying at least one program of instructions executable by the machine to perform a method of communicating between programs having different machine context organizations, said method comprising:

determining, at compile time, which savearea layout of a plurality of savearea layouts is to be used to save information relating to a calling program; and

selecting, at compile time, a linkage service from a plurality of linkage services to be used in communicating between said calling program and a callee program, wherein said calling program and said callee program coexist within a single executable module but have different machine context organizations.